TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

Viện Công nghệ thông tin và Truyền thông

Báo cáo giữa kỳ

Latent Semantic Analysis

IT4362 - Kỹ nghệ tri thức

Nhóm 2

Phạm Minh Khang (GL) - 20151955

Nguyễn Hoàng Dũng - 20150681

Võ Quốc Tuấn - 20152718

Nguyễn Duy Ý - 20154438

Hà Nội, ngày 3 tháng 11 năm 2019

Nội dung

Lời nói đầu

Trong những năm gần đây, khoa học và công nghệ đã và đang phát triển nhanh như vũ bão, đặc biệt là lĩnh vực công nghệ thông tin và khoa học máy tính. Với sự ra đời của máy tính điện tử đã thúc đẩy và đưa việc phát triển công nghệ lên tầm cao mới và mở khóa nhiều tiềm năng mới dành cho việc nghiên cứu và ứng dụng trong đời sống của con người chúng ta. Kỹ nghệ tri thức là một ngành tuy tuổi đời còn khá non trẻ khi so sánh cùng các ngành khác, tuy nhiên lại có tốc độ phát triển vô cùng ấn tượng, đạt được nhiều bước đột phá và cũng là định hướng tương lai của công nghệ loài người.

Hiểu được tầm quan trọng của môn học Kỹ nghệ tri thức, nhóm chúng em đã cố gắng và nỗ lực để học tập và tìm hiểu các kiến thức, triển khai các thuật toán bằng đề tài : Latent Semantic Analysis (LSA).

Mục tiêu của đề tài là bước đầu xây dựng một hệ thống Phân tích ngữ nghĩa tiềm ẩn qua trình xử lý ngôn ngữ thông qua các văn bản, bài báo.

Qua quá trình tìm hiểu kiến thức nền tảng, công nghệ và áp dụng vào thực tế, chúng em mới thực sự cảm nhận được những việc khó khăn khi xây dựng một hệ thống thông minh. Không chỉ là một quá trình đơn giản từ thiết kế tới thực hành, mà còn là các ý tưởng, thu thập thông tin, nhu cầu, xử lý đầu vào, đánh giá và nâng cấp mô hình, chạy thử nghiệm. Các công việc đều yêu cầu được sắp xếp hợp lý, khoa học và sự hợp tác của cả nhóm, cũng như sự hướng dẫn, giúp đỡ của các thầy cô và bạn bè. Chúng em xin được cảm ơn TS. Thân Quang Khoát đã định hướng và giúp đỡ kiến thức công nghệ, cảm ơn các bạn đã giúp đỡ trong quá trình phát triển và chạy thử nghiệm, cảm ơn Trường và ban lãnh đạo trường ĐHBKHN đã đưa bộ môn vào giảng dạy và tạo điều kiện cho chúng em thực hành và áp dụng.

Đề tài được xây dựng trên quá trình học tập và ứng dụng các công nghệ mới nên không thể tránh khỏi các hạn chế và thiếu sót. Kính mong thầy cô và các bạn góp ý, đánh giá và bổ sung để chúng em hoàn thiện kiến thức cũng như kinh nghiệm, cũng như đề tài có thể ứng dụng và hữu ích trong thực tế.

Chúng em xin chân thành cảm ơn !

I. Giới thiệu tổng quan

Phân tích ngữ nghĩa tiềm ẩn là một thuật toán trích xuất và đại diện nội dung ngữ nghĩa sử dụng tính toán thống kê với một tập văn bản lớn (Landauer và Dumais, 1997). Đây là một thuật toán đơn sử dụng đại số tuyến tính để biểu diễn tri thức, có nhiều áp dụng trong thực tế như xử lý nhiễu, trích chọn chủ đề, ..

Ý tưởng cơ bản là lấy tập hợp các từ trong tập văn bản, đưa ra từ xuất hiện hoặc không xuất hiện trong các văn bản thuộc tập văn bản đó,sau đó tính toán sự tương đồng của các từ với các từ khác hoặc củat ập từ với tập từ khác. LSA giả định rằng những từ có ngữ nghĩa gần nhau thường xuất hiện trong cùng ngữ cảnh. Xuất phát từ bảng dữ liệu D kích thước mxn, mỗi hàng tượng trưng cho một ký tự, mỗi cột tượng trưng cho một đoạn văn bản, mỗi một ô chứa tần suất mà từ ở dòng ma trận xuất hiện trong đoạn văn bản được biểu diễn tại cột của ma trận. Sau đó, LSA sử dụng kỹ thuật phân tích giá trị đơn (Singular Value Decomposition - SVD) rút trích mối tương quan ngữ nghĩa giữa các từ trong tập văn bản, giảm số cột (chiều) về k đặc trưng tiềm ẩn của bảng dữ liệu, thu được bảng R kích thước kxm trong khi vẫn giữ được cấu trúc tương tự của các dòng trong bảng R.

II. Kiến thức nền tảng

2.1 Kiến thức về ma trận

a) Trị riêng, vector riêng

Cho một ma trận vuông $A \in R^{n \times n}$, nếu số vô hướng λ và vector $x \neq 0 \in R^n$ thoả mãn: $\mathbf{Ax} = \lambda \mathbf{x}$ thì λ được gọi là một trị riêng của A và x được gọi là vector riêng tương ứng với trị riêng đó.

b) Hệ trực giao, trực chuẩn

Một hệ cơ sở $u_1, u_2, \ldots, u_m \in R^m$ được gọi là *trực giao* nếu mỗi vector là khác 0 và tích của hai vector khác nhau bất kỳ bằng 0:

$$\mathbf{u}_i \neq \mathbf{0}; \;\; \mathbf{u}_i^T \mathbf{u}_j = 0 \; \forall \; 1 \leq i \neq j \leq m$$

Một hệ cơ sở $u_1, u_2, \ldots, u_m \in R^m$ được gọi là *trực chuẩn* nếu nó là một hệ *trực giao* và độ dài Euclidean của mỗi vector bằng 1:

$$\mathbf{u}_i^T \mathbf{u}_j = \begin{cases} 1 & \text{if} \quad i = j \\ 0 & \text{otherwise} \end{cases}$$

Gọi $U = [u_1, u_2, \ldots, u_m]$ với $u_1, u_2, \ldots, u_m \in R^m$ là *trực chuẩn*, thế thì có thể suy ra :

$$\mathbf{U}\mathbf{U}^T = \mathbf{U}^T\mathbf{U} = \mathbf{I}$$

trong đó I là ma trận đơn vị bậc m. Ta gọi U là *ma trận trực giao* (orthogonal matrix)

c) Eigen-decomposition

Một ma trận vuông $A \in R^{n \times n}$ được gọi là *chéo hoá được* nếu tồn tại ma trận đường chéo D và ma trận khả nghịch P sao cho:

$$\mathbf{A} = \mathbf{PDP}^{-1} \quad \textbf{(1)}$$

Số lượng phần tử khác 0 của ma trận đường chéo D chính là rank của ma trận A. Nhân cả hai vế của (1) với P ta có:

$$\mathbf{AP} = \mathbf{PD} \quad (2)$$

Gọi p_i, d_i lần lượt là cột thứ i của ma trận P và D. Vì mỗi một cột của vế trái và vế phải của (2)phải bằng nhau, ta sẽ có:

$$\mathbf{Ap}_i = \mathbf{Pd}_i = d_{ii}\mathbf{p}_i \quad (3)$$

với d_{ii} là phần tử thứ i của p_i. Dấu bằng thứ hai xảy ra vì Dlà ma trận đường chéo, tức d_i chỉ có thành phần d_{ii} là khác 0. Biểu thức (3) chỉ ra rằng mỗi phần tử d_{ii} phải là một *trị riêng* của A và mỗi vector cột p_i phải là một *vector riêng* của A ứng với trị riêng d_{ii}.

Cách phân tích một ma trận vuông thành nhân tử như (1) còn được gọi là *Eigen Decomposition*.

2.2 Singular Value Decomposition

Mọi ma trận A có kích thước mxn được phân tích dưới dạng 3 ma trận như sau A=U.Σ.VT

-Ma trận U có kích thước mxm là một ma trận trực giao

-Ma trận V có kích thước nxn là một ma trận trực giao

-Ma trận Σ có kích thước mxn là một ma trận đường chéo

$$\sum_{m\times n} = \begin{bmatrix} D_{r\times r} & .... & 0 \\ ... & .... & ... \\ 0 & .... & 0 \end{bmatrix} \text{ với } D = \begin{bmatrix} \sigma_1 & .... & 0 \\ .... & .... & .... \\ 0 & .... & \sigma_r \end{bmatrix}$$

Các σi được gọi là các giá trị đơn (singular values) và $\sigma_1 \geq \sigma_2 \geq ....\sigma_r \geq 0$.

Các ma trận U, Σ, VT được xây dựng như sau:

- Các giá trị đơn $\sigma_i = \sqrt{\lambda_i}$ (λi là các giá trị riêng của ma trận ATA)
- Ma trận V được xây dựng dựa trên các vector riêng của ma trận ATA). Cụ thể: $V = [v_1 \quad .... \quad v_v]$
- Xây dựng ma trận U: Với các σi là giá trị đơn của ma trận A. Đặt $u_i = \frac{1}{\sigma_i} A v_i$ Từ đó xây dựng được ma trận $U = [u_1 \quad .... \quad u_m]$

Ví dụ: Triển khai SVD của ma trận $A = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

(1) Tìm các giá trị riêng của ma trận ATA. Ta có :

$$A^TA=\begin{bmatrix}1&0\\1&0\\0&1\end{bmatrix}\cdot\begin{bmatrix}1&1&0\\0&0&1\end{bmatrix}=\begin{bmatrix}1&1&0\\1&1&0\\0&0&1\end{bmatrix}$$

Giải phương trình det(A-λ.I)=0 được các giá trị riêng của ma trận ATA là $\lambda_1=2,\ \lambda_2=1,\ \lambda_3=0.$ Với mỗi giá trị riêng i giải phương trình $(A-\lambda I)x=0$ ta được các vector riêng tương ứng là:

$$v_1=\begin{bmatrix}1/\sqrt{2}\\1/\sqrt{2}\\0\end{bmatrix},v_2=\begin{bmatrix}0\\0\\1\end{bmatrix},v_3=\begin{bmatrix}-1/\sqrt{2}\\1/\sqrt{2}\\0\end{bmatrix}$$

Các giá trị đơn của ma trận A là: $\sigma_1=\sqrt{2},\sigma_2=1,\sigma_3=0$

Ma trận $\sum=\begin{bmatrix}\sqrt{2}&0&0\\0&1&0\end{bmatrix}$

Tìm ma trận U

$$u_i=\frac{1}{\sigma_i}Av$$

$$\Rightarrow u_1=\frac{1}{\sqrt{2}}\begin{bmatrix}1&1&0\\0&0&1\end{bmatrix}\begin{bmatrix}1/\sqrt{2}\\1/\sqrt{2}\\0\end{bmatrix}=\begin{bmatrix}1\\0\end{bmatrix}$$

$$u_2=\frac{1}{1}\begin{bmatrix}1&1&0\\0&0&1\end{bmatrix}\begin{bmatrix}0\\0\\1\end{bmatrix}=\begin{bmatrix}0\\1\end{bmatrix}$$

$$\Rightarrow U=\begin{bmatrix}1&0\\0&1\end{bmatrix}$$

Phân tích SVD của ma trận A là ：

$$A=U.\Sigma.V^T=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}\sqrt{2} & 0 & 0\\ 0 & 1 & 0\end{bmatrix}\begin{bmatrix}1/\sqrt{2} & 1/\sqrt{2} & 0\\ 0 & 0 & 1\\ -1/\sqrt{2} & 1/\sqrt{2} & 0\end{bmatrix}$$

Độ phức tạp của thuật toán SVD là O(n2k3), trong đó n là số từ, k là số chiều trong không gian.

2.3 Xấp xỉ ma trận, Truncated SVD

Trong thực tế, khi làm việc với các ma trận trong tính toán máy tính, ta thường phải đối mặt với các ma trận kích thước rất lớn (số chiều lên tới hàng triệu), khiến cho tốc độ và thời gian tính toán cũng như chi phí gia tăng. Áp dụng các thuật toán giảm chiều dữ liệu giúp giảm thời gian, chi phí nhưng vẫn giữ được chất lượng tính toán.

Giá trị của ma trận được thể hiện bởi Frobenius Norm. Bài toán xấp xỉ ma trận được đưa về bài toán tối ưu

$$\min_{\mathbf{B}} ||\mathbf{A}-\mathbf{B}||_F$$
$$\text{s.t. } \text{rank}(\mathbf{B})=k$$

Người ta đã chứng minh Truncated SVD là nghiệm tối ưu cho bài toán xấp xỉ ma trận rank k.

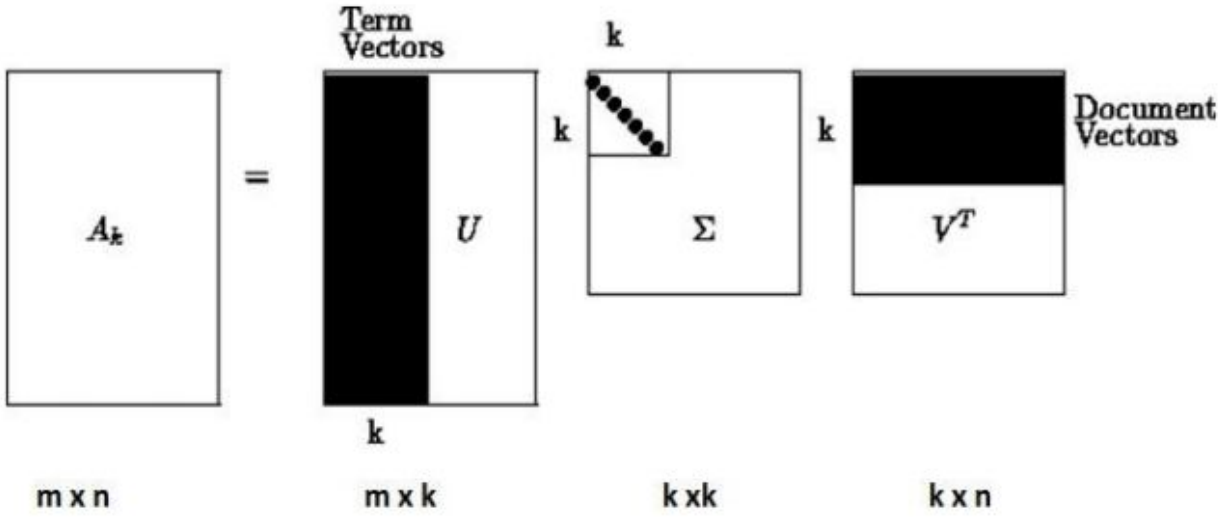
Để giảm số chiều của ma trận người ta thường tìm cách xấp xỉ ma trận A (có hạng r) bằng một ma trận Ak có hạng k nhỏ hơn rất nhiều.

Ma trận xấp xỉ của A là $A_k=U_k\Sigma_k V_k^T$, trong đó:

- Uk là ma trận trực giao m xk có các cột là k cột đầu của U.

- Σk là ma trận đường chéo k chứa các k phần tử đầu tiên σ1, σ2,...σk trên đường chéo chính.

- Vk là ma trận trực giao n k có các cột là k cột đầu của ma trận V.



Việc xấp xỉ này có thể xem như chuyển không gian đang xét (r chiều) về không gian k chiều, với k <<r.

III. Latent Semantic Analysis

Nền tảng của thuật toán được dựa trên ý tưởng biểu diễn trên không gian vector để tự động hoá quá trình trích xuất thông tin tự động. Để tính sự tương đồng giữa 2 văn bản, ta đánh giá dựa trên khoảng cách dựa trên biểu diễn của chúng.

Ta biểu diễn các văn bản dưới dạng Term-Document Matrix

	d_1	d_2	d_3	d_4	d_5
romeo	1	0	1	0	0
juliet	1	1	0	0	0
happy	0	1	0	0	0
dagger	0	1	1	0	0
live	0	0	0	1	0
die	0	0	1	1	0
free	0	0	0	1	0
new-hampshire	0	0	0	1	1

Trong thực tế, ta có nhiều cách để biểu diễn tập dữ liệu đầu vào, có thể bằng số lần xuất hiện hoặc để tăng kết quả, nhiều khi người ta sử dụng ma trận với trọng số global, local weighted hoặc biểu diễn ở dạng TF-IDF.

Sau khi biểu diễn tập dữ liệu ban đầu ở dạng ma trận, ta tiến hành áp dụng thuật toán Truncated SVD để tách ra 3 ma trận Uk, Sk và Vk. Lấy ví dụ là ma trận Term -Document ở trên, qua thuật toán SVD ta tách đuojc 3 ma trận U, S, V tương ứng

Ma trận U - Left Singular Matrix

-0.310866	-0.40733	-0.594461	-0.603046	-0.142814
0.362933	0.540742	0.200054	-0.695391	-0.228662
-0.118013	0.676704	-0.659179	0.198375	0.232971
0.860986	-0.28736	-0.358175	0.0530948	0.212177
0.128132	0.0342945	-0.209255	0.332558	-0.909958

Ma trận S - Singular Values

2.2853	·	·	·	·
·	2.01026	·	·	·
·	·	1.3607	·	·
·	·	·	1.11814	·
·	·	·	·	0.796577

Ma trận V -Right Singular Matrix

-0.396153	0.280057	-0.571171	0.449685	-0.101839
-0.314268	0.449532	0.410591	0.513018	0.203906
-0.17824	0.268992	0.497321	-0.256998	0.0430523
-0.438364	0.368508	0.0128792	-0.577329	-0.21964
-0.263881	-0.345921	0.145789	0.0474849	0.417484
-0.524005	-0.246405	-0.338652	-0.272846	0.154791
-0.263881	-0.345921	0.145789	0.0474849	0.417484
-0.326373	-0.459669	0.317003	0.237244	-0.724851

Uk là 2 cột đầu của U. Vk là 2 hàng đầu của V và Sk là 2 giá trị đầu của S.

Lấy tích Uk.Sk.Vk ta sẽ có ma trận Ak mới như sau.

0.485762	0.673199	0.65081	0.154457	0.00056003
0.551237	0.781199	0.607724	-0.195303	-0.104067
0.322878	0.45832	0.35032	-0.130389	-0.0654744
0.580283	0.808641	0.743726	0.0889825	-0.0263216
-0.0649144	-0.130389	0.219371	0.847233	0.245133
0.19249	0.219931	0.612777	1.0666	0.284286
-0.0649144	-0.130389	0.219371	0.847233	0.245133
-0.103507	-0.195863	0.258524	1.09237	0.317815

Để biểu diễn không gian từ và văn bản ta sử dụng các ma trận sau:

ma trận biểu diễn từ

$$U_2\Sigma_2$$

romeo	-0.905327	-0.562988
juliet	-0.718196	-0.903676
happy	-0.40733	-0.540742
dagger	-1.00179	-0.740797
live	-0.603046	0.695391
die	-1.19751	0.495337
free	-0.603046	0.695391
new-hampshire	-0.74586	0.924053

ma trận biểu diễn văn bản

$$\Sigma_2 S_2^T$$

d1	-0.710421 0.72959
d2	-0.930871 1.08703
d3	-1.35852 0.402161
d4	-1.37814 -1.39792
d5	-0.326373 -0.459669

. Sử dụng các phép đo khoảng cách để tính độ tương đồng. Với các query bất kỳ, biểu diễn của query được tính bằng centroid của các vector biểu diễn từ thành phần.

$$q = \frac{\sum_{i=1}^{N} q_i}{N}$$

Ví dụ : query = [die, dagger]

die = [-1.197 -0.494]
dagger = [-1.001 0.742]

q = ([-1.197 -0.494] + [-1.001 0.742]) / 2
= [-1.099 0.124]

Độ tương đồng giữa q và d

$$s = \frac{q.d}{|q||d|}$$

Một tính chất đặc biệt của LSA so với các phương pháp biểu diễn trước đó là khả năng biểu diễn các văn bản/ từ chưa biết - chưa xuất hiện/ mới và có thể cập nhật (updating/ downdating) .

Cách tốt nhất để thực hiện là tính lại từ đầu các hệ số và ma trận. Tuy nhiên điều này có thể tốn kém và không khả thi khi mật độ thay đổi quá cao. Một cách khác là tính toán dựa trên các biểu diễn sẵn có. Phương pháp được gọi là “fold - in” , biểu diễn tri thức của term và doc về không gian vector k chiều. Với các term mới, ta tính biểu diễn bằng công thức

$$t_n = t.V_k.\Sigma_k^-1$$

Tương tự với văn bản mới

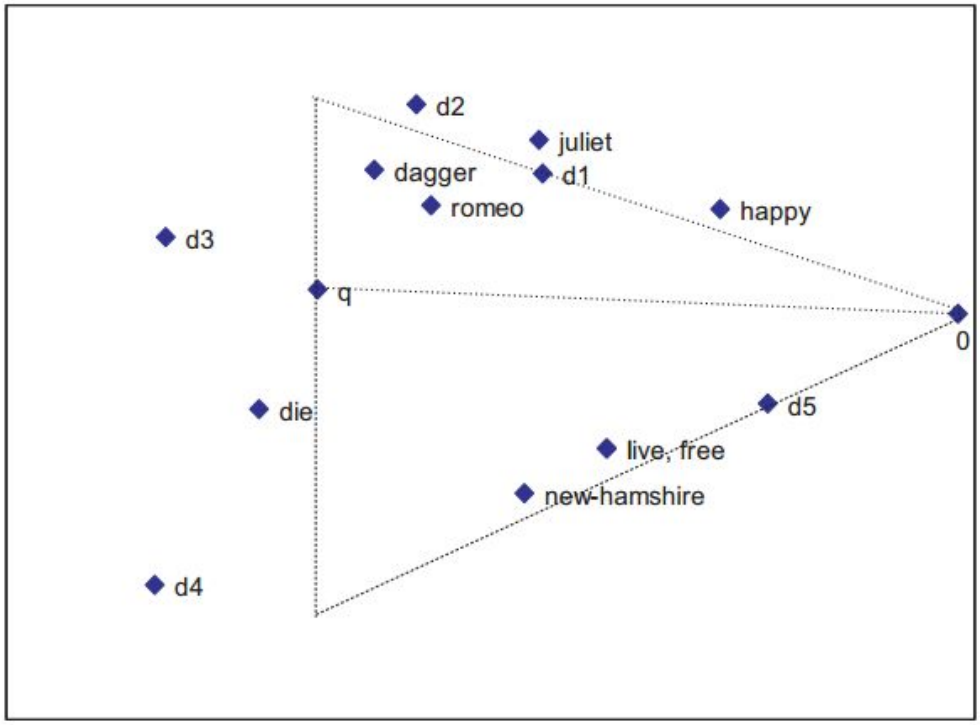
$$d_n = d^T.U_k.\Sigma_k^-1.$$

Trong trường hợp downdating, ta có thể coi như các từ/ văn bản không có mặt và coi như từ mới, hoặc loại bỏ trọng số của chúng khỏi các ma trận tri thức. Cách tốt nhất vẫn là tính lại các biểu diễn sau quá trình thay đổi.

IV. Triển khai thực tế/ Demo

4.1 Demo thuật toán

Diễn giải từng bước của thuật toán và chạy thử để xem kết quả của các bước. Nhóm sử dụng ngôn ngữ Julia/ Python để demo thuật toán và hiển thị các bước. Các ma trận được hiển thị và kết quả biểu diễn các từ:



4.2 Triển khai trên dữ liệu thực tế

a) Chuẩn bị và xử lý dữ liệu:

Dữ liệu được lấy từ github (https://github.com/duyvuleo/VNTC)

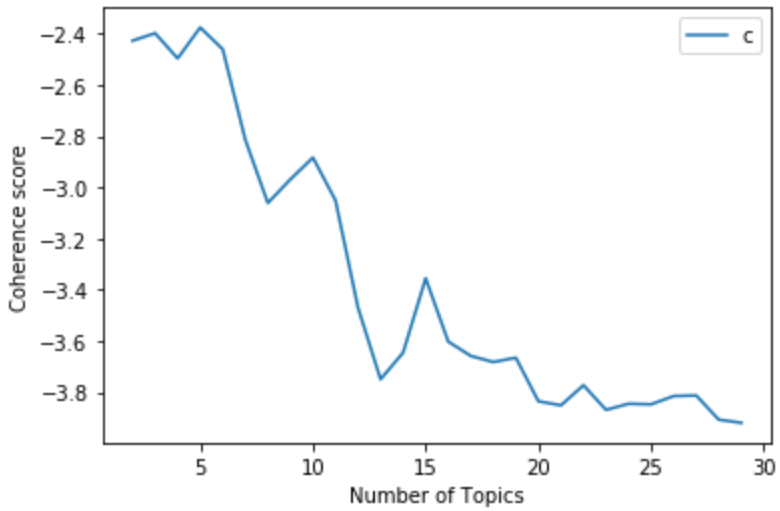
Các dữ liệu được dùng là 10Topics-ver1.1.

Các bước xử lý dữ liệu:

- Tách văn bản thành các tokens để xử lý
- Loại bỏ các từ không hợp lệ
- Loại bỏ các stopwords
- Tạo 1 bộ từ điển idx2term
- Tạo term-document matrix từ bộ dữ liệu

b) Lựa chọn mô hình

Sử dụng thư viện gensim - thư viện của python chuyên về topic models để thử nghiệm áp dụng thuật toán LSA/ LSI. Do mô hình phụ thuộc vào việc chọn số topic để giảm chiều, nhóm tạo 1 Coherence Model - tính mức độ liên quan giữa các từ trong topic để đánh giá và lựa chọn mô hình tốt nhất . Kết quả ra như sau:



Số topic tối ưu cho dữ liệu ban đầu là 5, sau khi tạo mô hình trả về kết quả như sau:

LSI Model:

Topic #0: 0.726*"nam" + 0.318*"hai" + 0.280*"thi" + 0.126*"doanh nghiệp" + 0.109*"hlv" + 0.102*"triệu" + 0.089*"usd" + 0.088*"kinh tế" + 0.086*"lan" + 0.083*"thi đấu"

Topic #1: -0.807*"thi" + 0.253*"hai" + -0.170*"hoc" + 0.139*"hlv" + 0.117*"nam" + -0.108*"tuyển sinh" + -0.107*"truong" + -0.105*"sinh" + -0.098*"khong" + 0.093*"thi đấu"

Topic #2: -0.527*"nam" + 0.450*"hai" + 0.269*"thi" + 0.238*"hlv" + 0.153*"chelsea" + 0.144*"pha" + -0.141*"doanh nghiệp" + 0.138*"chiến thắng" + 0.132*"thi đấu" + 0.123*"arsenal"

Topic #3: 0.902*"phim" + 0.184*"vai" + -0.113*"nam" + -0.104*"hlv" + 0.081*"usd" + 0.080*"thuốc" + 0.079*"triệu" + 0.075*"gia đình" + -0.073*"chelsea" + -0.073*"thi đấu"

Topic #4: 0.447*"thuốc" + 0.319*"usd" + -0.304*"phim" + 0.284*"triệu" + -0.249*"nam" + 0.194*"mua" + 0.176*"doanh nghiệp" + 0.170*"gia đình" + 0.143*"hai" + -0.142*"hlv"

c) Biểu diễn trực quan

Từ kết quả các topic ở trên, biểu diễn wordcloud cho các từ liên quan nhất của mỗi topic.

Topic 1

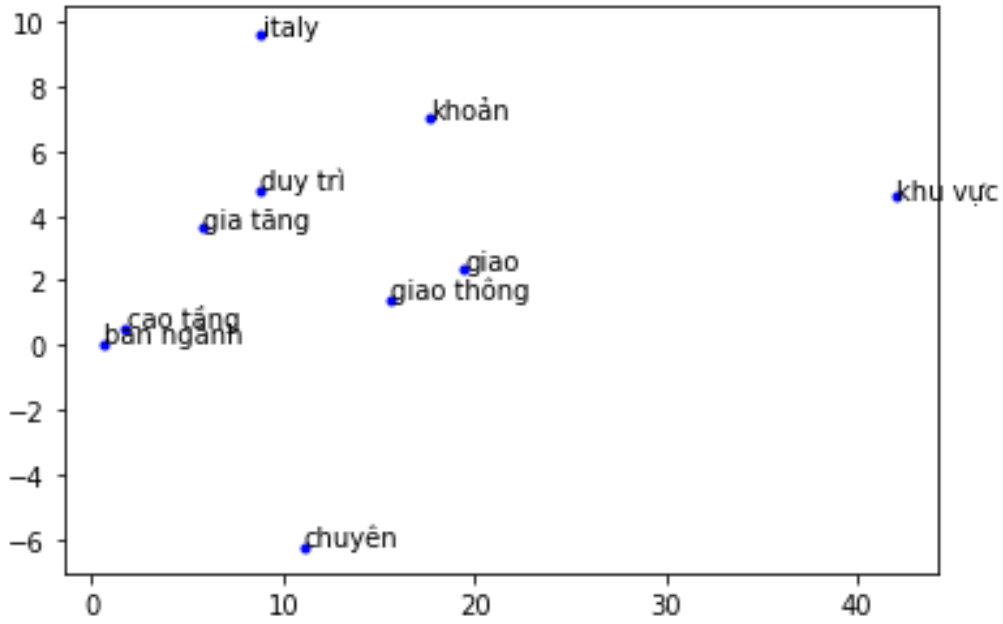
Topic 2

Topic 3

Topic 4

Topic 5

Ngoài ra, dữ liệu ban đầu có thể giảm về 2 chiều để biểu diễn không gian vector cho mỗi từ và mỗi văn bản. Đây là kết quả của 10 từ đầu tiên:



V. Đánh giá và kết luận

5.1 Đánh giá và nhận xét

- Điểm mạnh:
  - Mô hình đơn giản, hiệu quả
  - Khả năng ứng dụng cao
- Điểm yếu:
  - Cần nhiều dữ liệu để đạt hiệu quả tốt
  - Mô hình ngôn ngữ chưa hoàn thiện
  - Không biết chính xác topic là gì

5.2 Khả năng ứng dụng và tiềm năng

- Sử dụng nhiều trong Search-engine
- Sử dụng để giảm chiều , khử nhiễu

5.3 Kết luận

- Mô hình đơn giản nhưng hoạt động khá tốt
- Dễ dùng, dễ hiểu
- Có khả năng ứng dụng nhiều
- Mô hình ngôn ngữ chưa hoàn thiện

Phân công/ Đóng góp

Các nội dung demo, báo cáo, slide được lưu trữ và cập nhật trên link GitHub: https://github.com/kurokourou/LSA_demo

1. **Demo**

    Demo thuật toán (Sample demo) : Khang

    Demo

    + Chuẩn bị và tiền xử lý: Dũng
    + Chọn mô hình và đánh giá: Ý
    + Biểu diễn tri thức: Tuấn

2. **Báo cáo + Slide**

    Giới thiệu : Dũng

    Kiến thức nền tảng: Ý

    LSA + Edit: Khang

    Triển khai: Tuấn

Tham khảo

[1] Thomas K. Landauer, Susan T. Dumais, A Solution to Plato's Problem:The Latent Semantic Analysis Theory of Acquisition, Induction and Representation of Knowledge
http://lsa.colorado.edu/papers/plato/plato.annote.html

[2] Thomas K. Landauer, Danielle S. McNamara, Handbook of Latent Semantic Analysis, First Edition, Psychology Press

[3] Alex Thomo, Latent Semantic Analysis (Tutorial)
https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&cad=rja&uact=8&ved=2ahUKEwie64vhjNDlAhWMF4gKHVkaAKgQFjABegQIAhAC&url=https%3A%2F%2Fpdfs.semanticscholar.org%2F3efd%2Fa6e61747fea6b5cb5fa4f3ff0a14c86a638c.pdf&usg=AOvVaw0HGlDVpOS5yRNX-VMA1ilN

[4] Christopher D. Manning, Chapter 18: Matrix Decomposition and Latent semantic indexing, Introduction to Information Retrieval, First Edition, Cambridge University Press

[5] Machine learning cơ bản
https://machinelearningcoban.com/2017/06/07/svd/